
# VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

## FAKULTA STROJNÍHO INŽENÝRSTVÍ
FACULTY OF MECHANICAL ENGINEERING

## ÚSTAV AUTOMATIZACE A INFORMATIKY
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE


# ANIMACE LOGISTICKÝCH SYSTÉMŮ VE VÝROBĚ
LOGISTIC SYSTEM ANIMATION FOR PRODUCTION FACILITIES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS


AUTOR PRÁCE — Ing. Michal Mařata
AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. Simeon Simeonov, CSc.
SUPERVISOR



BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

# Zadaní bakalářské práce

| | |
|---|---|
| Ústav: | Ústav automatizace a informatiky |
| Student: | **Ing. Michal Mařata** |
| Studijní program: | Strojírenství |
| Studijní obor: | Aplikovaná informatika a řízení |
| Vedoucí práce: | **doc. Ing. Simeon Simeonov, CSc.** |
| Akademický rok: | 2020/21 |

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

## Animace logistických systémů ve výrobě

**Stručná charakteristika problematiky úkolu:**

Systémy pro animaci logistických systémů ve výrobě podporují:
– 3D vizualizace logistických a výrobních systémů.
– zvyšují atraktivitu prezentace výsledků simulace.
– vedou k urychlení analýzy řešených problémů.
– umožňují lepší rozvrhování výroby, apod.

**Cíle bakalářské práce:**

– Popis a porovnání systémů pro animaci logistických systémů ve výrobě.
– Naprogramování SW pro animaci logistických systémů ve výrobě.

**Seznam doporučené literatury:**

BRANDIMARTE P., Villa A. „Modeling Manufacturing Systems“ ISBN 3-540-65500-X, Springer, 2011.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2020/21

V Brně, dne

L. S.

| | |
|---|---|
| doc. Ing. Radomil Matoušek, Ph.D.<br>ředitel ústavu | doc. Ing. Jaroslav Katolický, Ph.D.<br>děkan fakulty |

# ABSTRAKT

Tématem práce je animace logistických systémů ve výrobě. Animace plánování výroby přidává možnost atraktivního, rychlejšího pochopení časového průběhu celého procesu. Výsledkem této práce je klasický program pro běh v prostředí Windows, psaný v programovacím jazyce C#. Pro zlepšení funkce 3D vizualizace je zvolena knihovna HelixToolkit. Dále je výstupem navíc i webová aplikace postavená na JavaScriptové knihovně Three.js. Aplikace umožňuje načtení vlastního 3D modelu výrobní haly (*.ifc soubor), načtení 3D modelů strojů a skladů, jejich přemístění a ukládání změn zpět do databáze. Součástí je demoverze se vzorovými daty, je možnost nahrát, podle předdefinované struktury, vlastní data (z MS Excelu, v desktop verzi i z MS Accessu). Během animace je umožněna možnost provádění změn (rychlost, přeskakování kroků).

# ABSTRACT

The topic of this thesis is Logistic Systém Animation for Production Facilities. Animation planning of production gets posibility of more atractive, faster understanding of timing production line. The result of the thesis is classic program for Windows, written in C#. For better functionality of 3D visualization is addedd HelixToolkit library. The other result is web application written with JavaScript library Three.js. In Web Application is enabled to load 3D model of own factory (*.ifc file), loading other 3D models, changing their position and saving changes to the database (MS Excel, MS Access). During the animation is enabled to make changes (as speed, skipping steps).

# KLÍČOVÁ SLOVA

animace logistických systémů, 3D vizualizace, rozvrhování výroby, plánování výroby, simulace výroby, výrobní linka, WebGL, HTML5, PHP, MySQL, CSS, javascript, Three.js, C#, Helix Toolkit

# KEYWORDS

animation of logistics systems, 3D visualization, production scheduling, production planning, production simulation, production line, WebGL, HTML5, PHP, MySQL, CSS, javascript, Three.js, C#, Helix Toolkit

# BIBLIOGRAFICKÁ CITACE

MAŘATA, Michal. *Animace logistických systémů ve výrobě.* Brno, 2021. Dostupné také z: https://www.vutbr.cz/studenti/zav-prace/detail/132022. Bakalářská práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav automatizace a informatiky. Vedoucí práce Simeon Simeonov.

# PODĚKOVÁNÍ

Tímto bych rád poděkoval panu doc. Ing. Simeonu Simeonovi, CSc. za vypsání zajímavého tématu v oboru aplikované informatiky, poskytnutí podkladů ke zpracování bakalářské práce a za cenné rady a připomínky při tvorbě této práce.
Dále bych chtěl poděkovat panu doc. Ing. Janu Roupcovi, Ph.D. za cenné rady ohledně tvorby bakalářské práce v rámci Semináře k bakalářské práci.

# ČESTNÉ PROHLÁŠENÍ

Prohlašuji, že tato práce je mým původním dílem, zpracoval jsem ji samostatně pod vedením doc. Ing. Simeona Simeonova, CSc. a s použitím literatury uvedené v seznamu literatury.

V Brně dne 21. 5. 2021 ……

Michal Mařata

# OBSAH

# 1 ÚVOD

Tato bakalářská práce se zabývá návrhem jednoduchého softwaru a navíc i webové aplikace pro animaci výroby pomocí 3D počítačové vizualizace. Možnost animace průběhu produktu výrobní linkou je výhodná pro efektnější prezentaci postupu výroby, rychlejší analýzu řešených obchodních problémů.

V následující kapitole stručně popíšeme, co si v kontextu této práce představit pod pojmem logistický systém ve výrobě, plánování výroby a zamyslíme se nad důvody, zda a proč má smysl zabývat se právě i animací výroby.

V další kapitole se stručně seznámíme s některými ze stávajících obchodních řešení. Mezi nimi najdeme jak robustní aplikace, které splňují nejnáročnější požadavky na 3D vizualizaci, tak i takové, které jsou výpočtově sice velmi dobře propracované, nicméně na dnešní dobu už méně graficky zajímavé. V neposlední řadě se lze v praxi setkat i s případy, kdy se využívá jen oblíbený tabulkový procesor.

Postupně se tak přesvědčíme, že je na trhu stále prostor pro vytvoření aplikace, která umožní využít exporty ze stávajících firemních „excelových“ tabulek, nebo lépe z plánovacích programů a tyto použít jako zdroj dat pro vlastní aplikaci sloužící k animaci plánování výroby. Dostaneme tak nástroj, který splní očekávání dnešní doby na vizualizaci ve 3D a přitom navrhované řešení v této práci umožní firmám ponechat si stávající plánovací nástroje, se kterými umí pracovat. Použití 3D grafické nástavby na 2D stávající programy dává smysl nejen pro hezčí a modernější prezentaci, avšak v případě složitějších systémů výrobních linek ve více úrovních, kdy cesty dopravníků vedou různě nad jednotlivými stroji, pak i praktický smysl, protože zobrazení ve 2D režimu zde není dostatečně jednoznačné.

Navržen je jak standardní desktopový program, což je zadaný cíl práce, tak i webová aplikace. Ta vznikla navíc v reakci na současné potřeby klientů, kteří stále častěji vyžadují přístup k informacím odkudkoli, z jakéhokoli mobilního zařízení. Tomuto předpokladu nejvíce vyhovuje právě webová aplikace.

Obě verze umožní nahrání výrobního plánu z excelového souboru do databáze aplikace nebo programu. Webová aplikace navíc oproti desktopové verzi umožní načtení modelu výrobní haly, nezbytností jsou možnosti úprav, jako správné rozmístění strojů po ploše výrobní haly jednoduchým „drag&drop“ způsobem.

Aplikace ani program prozatím necílí na nahrazení a konkurování stávajícím obchodním řešením, což by bylo nad rámec stanoveného rozsahu bakalářské práce a dosud získaných programovacích dovedností. Stávající programy jsou jistě mnohem lépe propracované, s větší škálou programových funkcí, graficky ještě více atraktivnější. Pro větší společnosti budou tyto stávající obchodní řešení určitě nezbytností, pro menší podniky by však zamýšlené řešení v této bakalářské práci mohlo být zajímavé z výše popsaných důvodů.

# 2 PLÁNOVÁNÍ VÝROBY

Plánování výroby je nezbytné pro efektivní využívání zdrojů výroby, zajištění včasného naskladňování polotovarů, splnění termínů expedice, dodání hotových výrobků klientovi.

V této kapitole bude ukázáno, že plánování výroby hraje ve výrobním procesu významnou roli, přestože je mnohdy opomíjeno. Určitě je možné plánovat výrobu bez nutnosti vytvářet 2D či 3D animace, simulace výroby. Na druhou stranu, stejně jako například v oboru konstruování, má využívání moderní výpočetní techniky určitě význam i v oblasti plánování.

V dnešní době se nikdo nepozastavuje nad tím, proč vlastně projektovat, konstruovat v CAD nástrojích, většinou dnes už právě v podobě 3D počítačového modelování. Výhody takového přístupu jsou zřejmě všem jasné. I přes počáteční větší úsilí a časovou náročnost na vytvoření 3D modelu, nám v dalších fázích vývoje konstrukce přijde vhod, že je možné 3D model už poměrně jednoduše upravit, zapracovat případné nové přání klienta, znovu už celkem jednoduše vygenerovat všechny potřebné řezy, pohledy, výpisy materiálu.

V případě vytváření plánu výroby se jedná o podobný vývoj změny pracovních zvyklostí, kdy z vytváření různých harmonogramů v papírové podobě se přecházelo na využívání specializovaných programů, pro které se zde zažil název APS – Advanced Planning and Scheduling Systems. I kdyby menší společnosti nepoužívali ještě tyto programy, určitě nevytváří harmonogramy v ruce a využívají alespoň oblíbeného tabulkového procesoru MS Excel, programu pro harmonogramy MS Project a jim podobné alternativy.

Stejně jako se i ve zmíněném příkladu z oboru konstruování postupně přešlo až k projektování na základě 3D počítačových modelů, stejně tak tomu postupně bude i u plánování výroby.

Mezi hlavní cíle správného plánování patří zejména:

- minimalizace nákladů na výrobu produktu,
- výroba maximálního počtu výrobků za nejkratší čas při využití současných zdrojů,
- případně nalezení tzv. úzkých míst,
- jejich odstranění optimalizací výroby,
- dosažení maximálního zisku.

Plánování výroby lze pak rozdělit například na následující oblasti:

- Strategické plánování - plánování a výstavba nových výrobních kapacit
- Taktické plánování - tvorba výrobních plánů na základě poptávky, alokace zdrojů
- Operativní plánování – rozvrhování a řízení výroby [1]

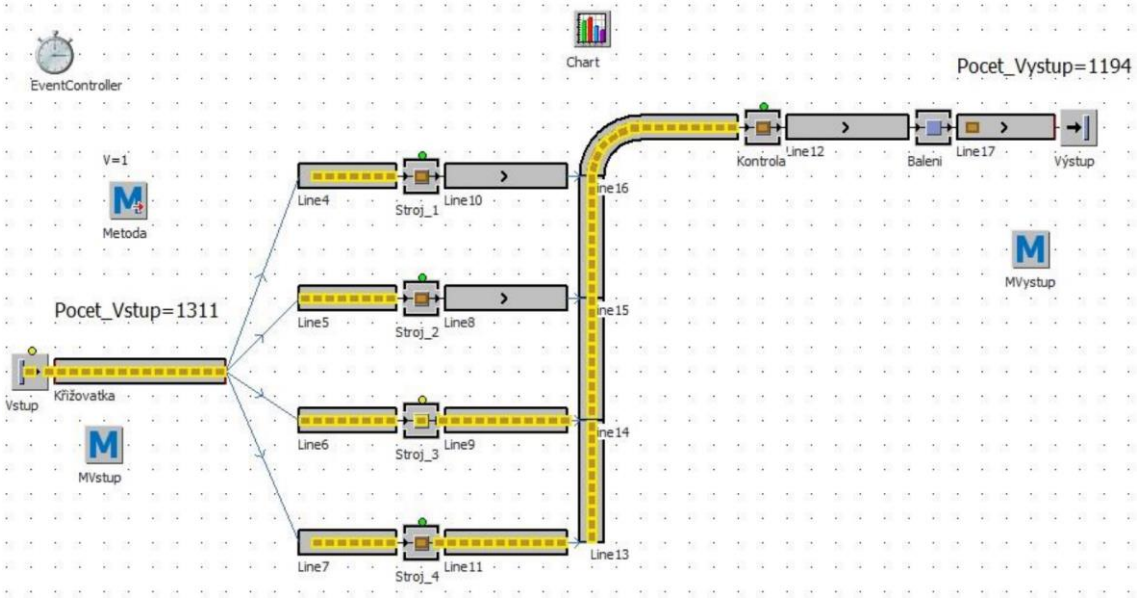
V rámci tématu této práce nás bude zajímat zejména oblast operativního plánování.

**Logistický, výrobní systém**

Pod pojmem logistický systém, ať už výrobní, nebo obecný, si lze představit obecně soubor prvků, mezi nimiž existují vzájemné vazby. [2]

Aby výroba splnila se zákazníkem domluvené termíny, je potřeba její průběh umět naplánovat ideálně s podrobností denního plánu. Častým problémem neefektivně naplánované výroby jsou tzv. **fronty práce**, kdy stroj nestíhá zpracovávat výrobek tak rychle, jako předchozí stroje výrobního systému. Vytváření fronty práce, a tedy blokování výrobku u určitého stroje, lze vyčíst i z klasických grafických výstupů.

Nicméně na následujícím obrázku je vidět, že animace simulace výrobního cyklu nám dá názornější a rychlejší představu, než pouze z grafu, kde je problém tzv. „úzkého místa“.




Obr. 1: Ukázka namodelované fronty práce vzniklé kvůli nestíhajícímu „stroji kontrola“ [2]

**APS – Advanced Planning and Scheduling Systems**

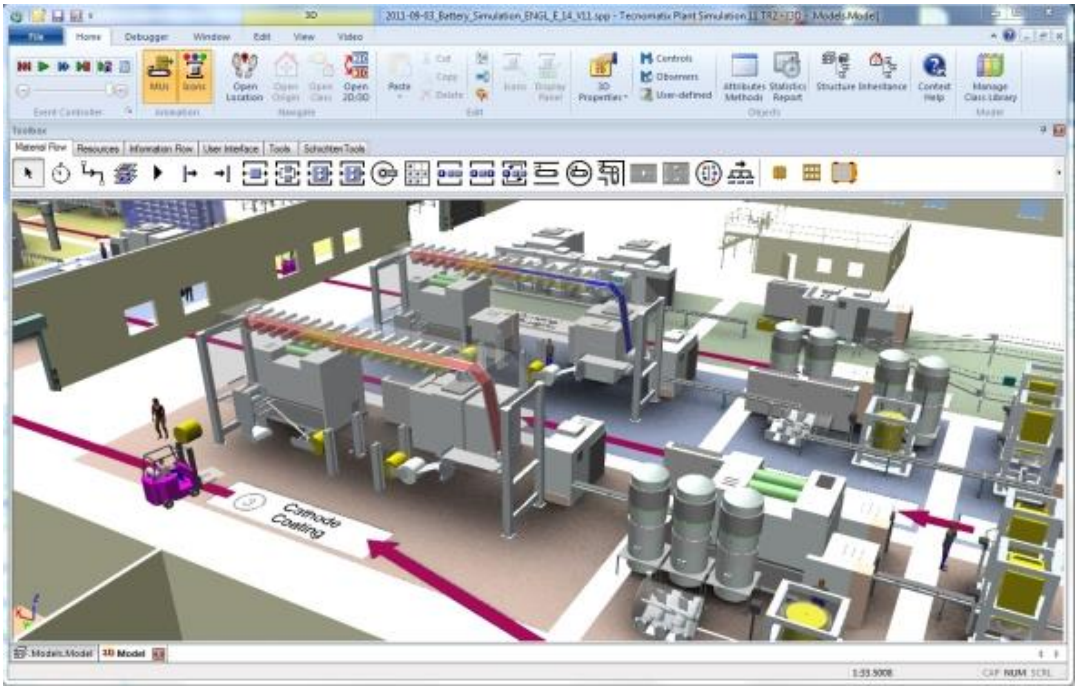
Díky plánování výroby tyto situace dokážeme výpočtovými metodami, či simulačními modely v případě složitějších systémů, odhalit a optimalizujeme výrobu nasazením vhodnější kombinace stroje, posílením kapacity meziskladů, regálů. K naplánování výroby dnes máme širokou škálu programového vybavení, pro něž se zažila zkratka APS – Advanced Planning and Scheduling Systems.

Jedná se o rozšířené programové nástroje pro dokonalejší plánování výroby, nalezení optimálních stavů výroby, s možností zadat reálné vstupní parametry, jako jsou časy dodávek, průběžné změny. APS postupně přicházely v 90. letech minulého století.

Nyní se pojďme přesvědčit, zda má smysl provádět i 3D animaci logistických systémů využíváním modernějších programových nástrojů. V případě jednodušších výrobních linek, těžko můžeme tvrdit, že animace a vizualizace ve 3D nám umožní lépe a rychleji odhalit úzká místa. Je tomu spíš naopak. Půdorysný pohled je přehlednější,

zobrazení ve 3D je náročnější, takže i případná odpověď počítače by mohla v případě starší verze být pomalejší.

Jiná situace nastane u složitějších systémů, kdy je výrobní linka řekněme víceúrovňová, dopravníkové pásy vedou nad jednotlivými stroji, jak je vidět například v následujícím ukázkovém obrázku modelu haly. Zde si s 2D zobrazením vystačíme hůře, i když odpůrci 3D projektování mohou namítnout, že i to lze samozřejmě v půdorysném průmětu řešit, umístěním druhého patra vedle a doplněním o příslušný odkaz detailu. Avšak zde už vidíme smysl a potenciál projektování ve 3D režimu.



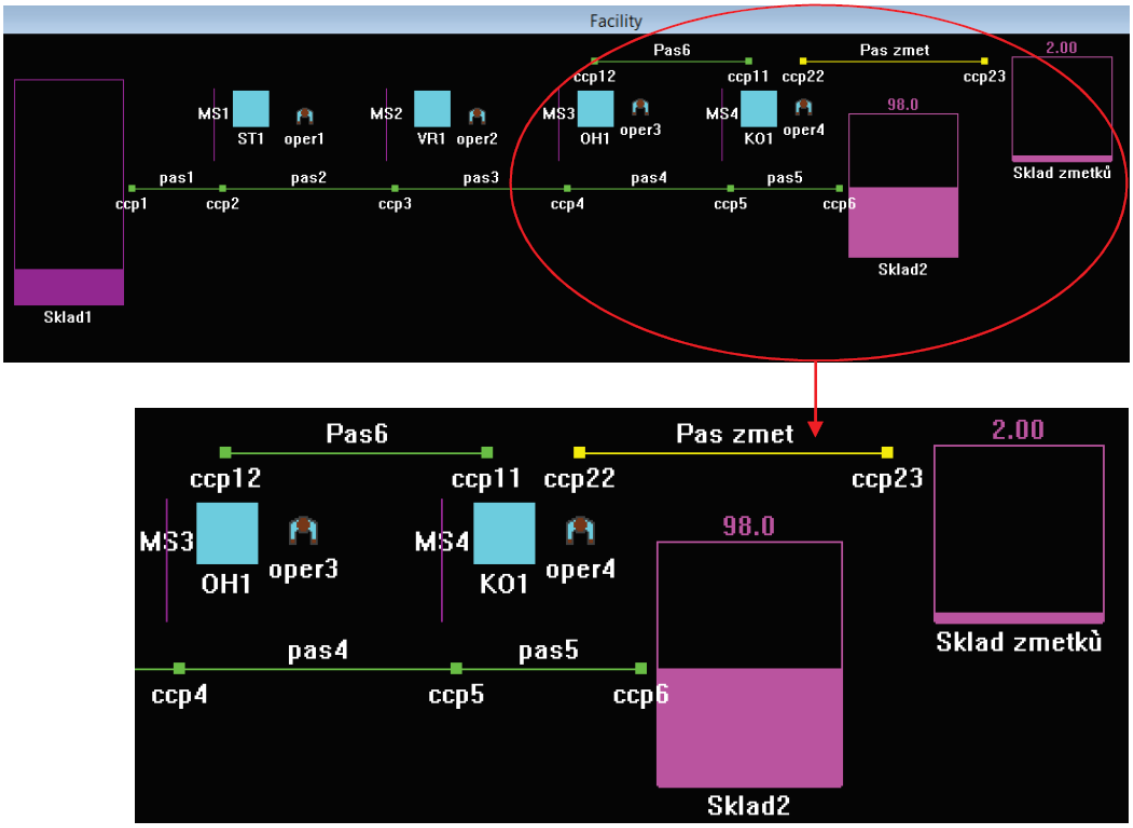

Obr. 2: Ukázka namodelované „digitální továrny“ v Siemens PLM softwaru [3]

Další důvod je také zřejmě samozřejmý. Dnešní doba si to vyžaduje. Stejně jako se říká, že obal výrobku prodává, tak i graficky přívětivější uživatelské prostředí lépe prodá nabízený software.

Příkladem běžně používaného systému pro výrobní plánování je také program **Factor/AIM**, který nám umožní navržení logistického, výrobní systému, provede simulaci, z grafických či textových výstupů nám dodá veškeré možné informace o využívání jednotlivých strojů. Na základě těchto dat vyhodnotíme, kde je úzké místo, provedeme optimalizaci upravením návrhu.

Slabinou tohoto programu však je, jakým způsobem lze prezentovat probíhající simulaci. Jak je vidět v ukázkách z programu, modeluje se většinou pomocí dostupných schématických nástrojů, ve 2D zobrazení. V dnešní době taková prezentace nezaujme, jelikož nynější programovací nástroje a webové technologie umožňují 3D vizualizace, což zákazník stále častěji vyžaduje. Čím více se simulační model přibližuje reálnému prostředí výrobní haly, tím lépe je takový produkt prodejný. Pokud zákazník není zaujatý

při prvotním seznámení se s novým programem, většinou se pak nepropracuje do jeho hloubky a tím pádem ani nedocení případné jiné kvality programu.




Obr. 3: Ukázka modelování výrobního procesu v Factor/AIM [4]

Tento ukázkový obrázek je vybrán záměrně, samozřejmě je možné, že i v rámci programu Factor/AIM si dáme práci s hezčím vizualizačním layoutem, ostatně i v některé z demo ukázek se to zdařilo. Nicméně to není v praxi běžné.

V rámci zadání tohoto tématu bakalářské práce je tak brán v potaz předpoklad, že zákazník si je už vědom výše popsaných důvodů, proč se vůbec zabývat plánováním. Využívá sice k rozvrhování své výroby nějaký výpočetní nástroj, jako je Excel, nebo specializovanější plánovací program, který mu umožňuje exportovat právě do formátu excelového souboru, ale na druhou stranu má omezené grafické možnosti pro vizualizaci pracovní plochy a vlastní animaci logistiky výroby.

Pro plánování výroby tak bude moct využívat své stávající pracovní metody, jako je dosud zvyknutý. Výsledky si pouze vyexportuje do excelového souboru, případně přeformátuje do struktury dle naší šablony. Takto upravený soubor použije jako zdroj dat, soubor jednoduše načte v našem softwaru či aplikaci navržené v rámci této bakalářské práce a výsledkem je, že si zvýší grafickou úroveň simulace, chceme-li animace výroby, případně lépe odhalí i nějaké chyby v plánování, kterých si nemusel všimnout v klasických tabulkových výpisech a grafech, jak bude ukázkově také probráno v kapitole vlastního řešení.

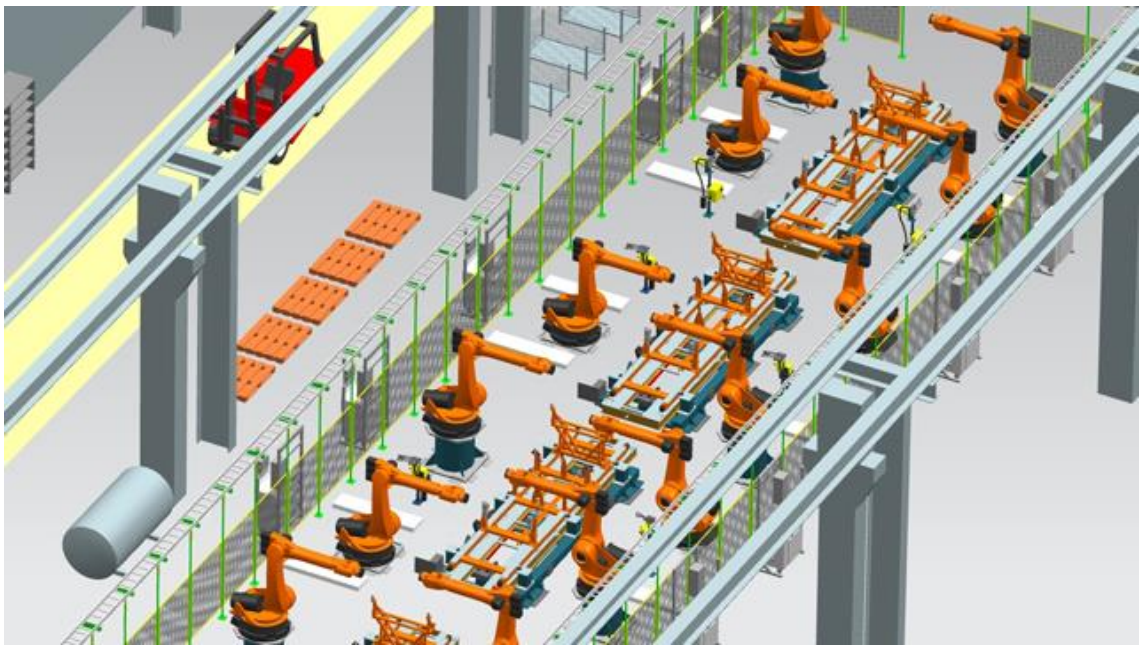
# 3 PŘEHLED SOUČASNÝCH OBCHODNÍCH ŘEŠENÍ

Po provedení průzkumu stávajících obchodních řešení pro simulaci výroby lze doporučit následující softwarové produkty pro tzv. diskrétní simulaci.

## 3.1 Tecnomatix Plant Simulation

Jedná se o komplexní řešení, které umožní vytvořit tzv. „digitální továrnu“, neboli také „digitální dvojče“ vlastní továrny, výrobní haly. Díky přesnému modelování, 3D vizualizaci, lze předcházet případným kolizím, které by jinak byly odhaleny až v průběhu výroby. Kromě samotné simulace výroby lze optimalizovat toky materiálu, využívání zdrojů a celkovou logistiku.

SIEMENS



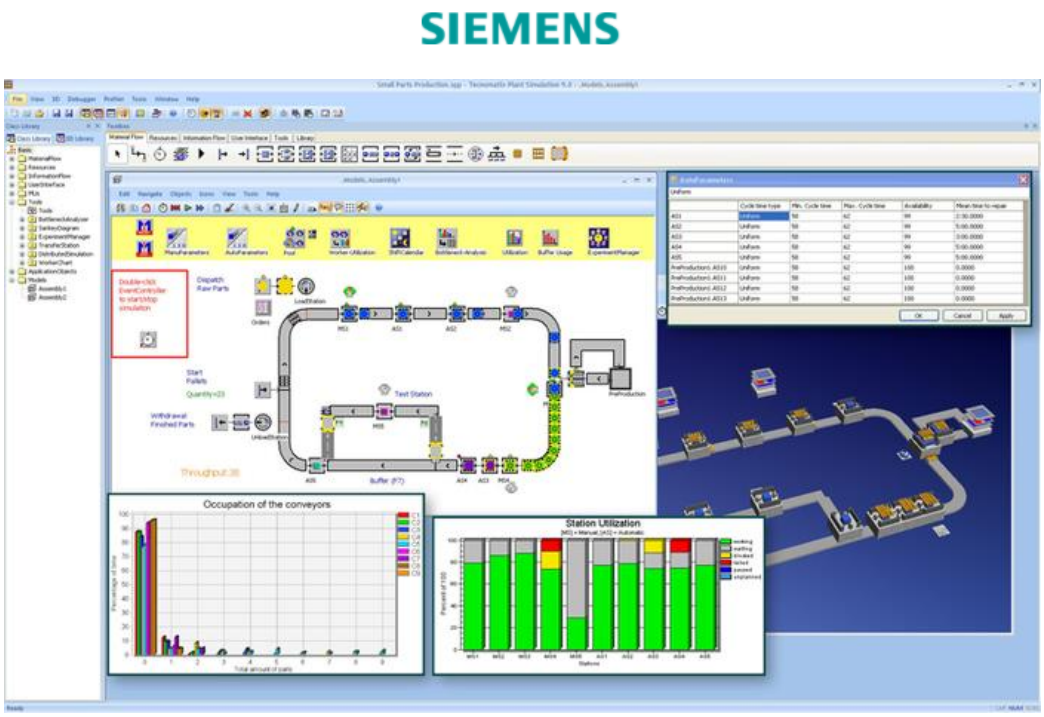
Obr. 4: Ukázka z programu od Siemens [5]

Vlastnosti programu:

- 2D a 3D vizualizace
- Rozsáhlé uživatelské knihovny
- Přizpůsobení pomocí programovacího jazyka SimTalk (varianta C++)
- Mnoho možností analýzy – Sankey diagram, Ganttův diagram (harmonogram)
- analýza úzkých míst
- Genetický algoritmus pro automatickou optimalizaci parametrů systému
- Odpovídá na otázky typu „Co se stane, když“ (metoda „What-if“)
- Umožňuje import dat z jiných systémů, jako MS Access, MS Excel.

Udává se, že správným používám programu lze:

- Zvýšit produktivitu stávajícího systému až o 20%
- Snížit investiční náklady při plánování nového systému také až o 20%
- Snížit zásoby na sklad o 20 – 60%
- Optimalizovat velikosti systému (haly), vč. velikosti skladů
- Snížit investiční riziko včasným ověřením průběhu výroby simulacemi
- Maximálně využít výrobních zdrojů
- Zkrátit dobu náběhu výroby




Obr. 5: Ukázka programu od Siemens [6]

## 3.2 FlexSim

Zajímavým softwarem, u nás méně komerčně prosazovaným, se zdá být FlexSim.
Jedná se o produkt od FlexSim Software Products, Inc., psaný v C++ pro operační systémy Windows 7 a vyšší. Disponuje rovněž množstvím standardních knihoven strojů, dopravních zařízení. Je objektově orientovaný. Model výrobního systému může být sestaven téměř bez nutnosti psaní programového kódu, pouze výběrem objektu, posunem metodou „drag&drop“ po pracovní ploše, nastavením vlastností v dialogovém, informačním okně o objektu, dostupném u většiny z nich. Zkušenější uživatelé mají nicméně možnost upravit vlastnosti a chování použitím programovacího jazyka FlexScript a C++.

Mezi dostupnými objekty jsou i robotická ramena s přednastavenou logikou pohybu a možností vytvořit vlastní cestu pohybu. Jako další lze zmínit jeřábové dráhy.

Disponuje možností různých exportů. V rámci postupného přechodu na tzv. Průmysl 4.0 má tento program určitě velký potenciál.

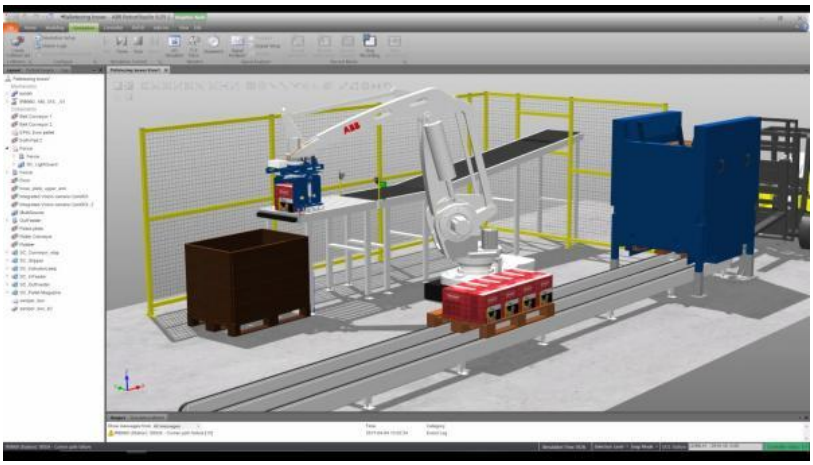

Obr. 6: Ukázka z programu FlexSim [7]

## 3.3 Arena Simulation Software

Jedná se o software od Rockwell Automation. Tvorba výrobního modelu sestává z vkládání modulů prezentujících stroje, sklady, palety a poté se spojovacími čarami propojí tak, aby vytvořily požadovaný tok výrobního procesu. Lze uživatelsky přizpůsobit a automatizovat pomocí programovacích jazyků SIMAN, Visual Basicu, C++. Rovněž umožňuje export a import MS Excelu a MS Accessu. [2]

## 3.4 ABB Robot Studio

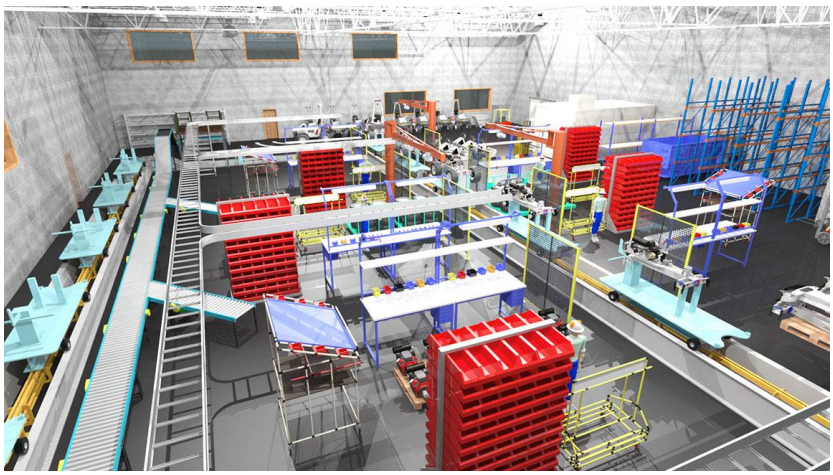
ABB Robot Studio je jeden z nejrozšířenějších nástrojů pro vytváření tzv. „digitálních dvojčat“ robotů, které jsou stále častěji nezastupitelnými stroji v rámci výrobní haly. Zejména proto je tento specializovaný software zařazen zde mezi popisem stávajících obchodních řešeních. Umožňuje nastavit a ověřit chování strojů/robotů bez nutnosti zastavování výroby, což mimo jiné vede ke zvýšení ziskovosti výroby, snížení rizik z případných chybných manipulačních kroků v rámci učení se nového výrobního postupu. Umožňuje velmi realistickou simulaci.



Obr. 7: Ukázka programu od ABB [8]

## 3.5 Autodesk – Factory design utilities

V neposlední řadě i společnost Autodesk, zřejmě lídr v oblasti počítačové podpory projektování a konstruování, má mezi svými produkty i software pro animaci výroby.



Obr. 8: Ukázka z programu od Autodesk [9]

## 3.6 Witness

Program od společnosti Lanner Group Ltd. Kombinuje diskrétní modelování se spojitým. Je tak vhodný nejen pro strojírenskou výrobu, ale i pro jiné obory. Spojité modelování procesu umožňuje simulovat prvky pohybující se velkými rychlostmi. Disponuje 2D i 3D vizualizací, importem dat z MS Excel, dokonce i CAD. Pro programování logiky chování prvků se používá jazyk Witness, který lze doplnit kódy externích knihoven z C++, C#, VB.net. Dále lze zmínit, že umožňuje změnu modelu během simulace. Vhodné rovněž nejen pro simulace výroby, ale například i provoz letiště. [10]

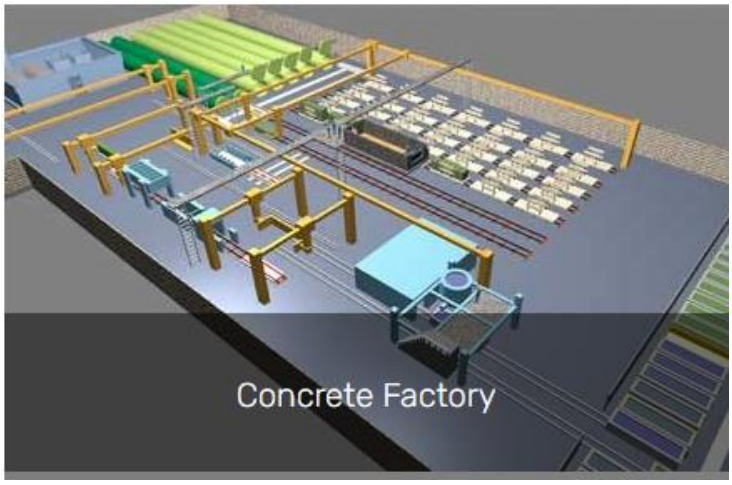
## 3.7 Factor/AIM

Tento program je využíván v rámci výuky Simulace výroby. Student se seznámí s tvorbou modelu výrobního systému, lze vložit moduly jako stroje, sklady, dopravníky, jeřáby, transportéry. Simulace probíhá velmi dynamicky, poskytuje množství výstupů, jako textový tabulkový výstup využití strojů, sloupcové grafy využití, Ganttovy diagramy (řádkové harmonogramy). Výstupem je databázový soubor kompatibilní s MS Accessem.

## 3.8 Simprocess

Program spojující diskrétní simulaci, s možností kalkulace nákladů. Umožňuje vlastní přizpůsobení logiky modelu, objektově orientované programování (jazyk JAVA). Modelování metodou „drag&drop“, obsahuje „What-if“ analýzy, optimalizace procesů. Standardní výsledné reporty, grafy, s možností exportu do PDF, RTF, PPT, HTML, XLS, databáze. Využívá se i k řízení obchodních procesů, call center.

## 3.9 Anylogic




Obr. 9: Další možné řešení od Anylogic [11]

# 4 POPIS NÁVRHU VLASTNÍHO ŘEŠENÍ

Tato část práce je zaměřena na popis vlastního řešení. Od vedoucího práce bylo obdrženo zadání, vč. ukázkové diplomové práce z roku 2012 od Ing. Veselského [12]. Součástí této diplomové práce jsou testovací databáze, soubory z MS Accesssu, které slouží jako zdroj dat pro vlastní vizualizaci a animaci průběhu výroby v čase.

Vybraná testovací databáze byla použita jako zdroj dat i pro tuto bakalářskou práci, s tím, že vytčeným cílem je přiblížit se funkčně a designově navrženému programu ve zmíněné diplomové práci, avšak za používání současných modernějších programovacích nástrojů a technologií. Tím je myšleno zejména pracování ve 3D režimu, což diplomová práce zpracovaná v té době ve své praktické části neumožňovala a zaměřila se na propracovanou animaci ve 2D zobrazí. Nutno podotknout, že o možnostech 3D animace však pojednávala v teoretické části.

Naproti tomu tato bakalářská práce je zaměřena výhradněji na praktickou část, popis případných úskalí při programování podobného řešení a neklade si za cíl široce pojednávat o teorii k použitým programovacím jazykům a technologiím.

Přestože hlavním cílem bakalářské práce je naprogramování software pro animaci logistických systémů, v dnešní době jsou i běžné desktopové aplikace postupně vytlačovány webovými aplikacemi přístupnými odkudkoli. V této práci se tak nejprve zaměříme na možnosti animace a vizualizace ve webových aplikacích. Jak bude ve výsledku ukázáno, i vzniklá webová aplikace může nahrazovat klasický software, přičemž si může zachovat podobný vzhled jako desktopový program.

V další kapitole pak získané znalosti z návrhu webové aplikace využijeme při tvorbě druhého vlastního řešení, softwaru, který bude spustitelný na počítačích s Windows, a to jako klasicky spustitelný *.exe soubor.

## 4.1 WEBOVÁ VERZE

V dnešní době je žádoucí mít přístup ke svým datům a aplikacím odkudkoli a na jakémkoli zařízení. Nejjednodušším řešením, jak spustit aplikaci odkudkoli a téměř zaručeně na jakémkoli zařízení, je naprogramovat webovou aplikaci. Webová aplikace je programována klasicky pomocí html kódu, JavaScriptové grafické knihovny pro 3D zobrazení, konkrétně v tomto našem případě **THREE.js**, využívající technologii **WebGL** pro 3D vykreslování. Pro přístup do **MySQL** databáze je dále používán jazyk **PHP**.

Po přihlášení se dostáváme k úvodní stránce aplikace, kde mimo jiné zvolíme ze dvou režimů, a to buď demoverze, anebo využije možnost nahrát přímo vlastní data.


Obr. 10: Ukázka úvodní obrazovky vlastní webové aplikace

Po otevření nového okna se zobrazí už samotná webová aplikace s 3D grafickým rozhraním, které lze poměrně rychle intuitivně ovládat.


Obr. 11: Ukázka vlastní webové aplikace

### 4.1.1 Ovládání aplikace:

Ovládání animace je navržené s maximální snahou o uživatelskou přívětivost, minimum zobrazených ovládacích tlačítek.

Před spuštěním máme možnost vybrat z nabízených možností rychlost animace, časový úsek animace. Animaci zahájíme zeleným tlačítkem **„SPUSTIT“** v horní nástrojové liště.

Po spuštění se v rámci uživatelské přehlednosti během animace zneviditelní aktuálně nepotřebná ovládací tlačítka. Samozřejmostí je možnost přerušení animace barevně vhodně zvoleným červeným tlačítkem **„PŘERUŠIT“.**

Kliknutím na tlačítko **„Možnosti“** se zpátky zobrazí celá nástrojová lišta, kde opět můžeme např. upravit rychlost animace a následně obnovit simulaci kliknutím na tlačítko **„POKRAČOVAT“.**

Co se týká ovládání vizualizační plochy, je dáno zejména možnostmi použité knihovny Three.js. Je velmi intuitivní a dodržuje běžné standardy technikům známé z CAD nástrojů. Držením levého tlačítka myši a současným posunem myši lze 3D model pomyslné výrobní haly různě natáčet, běžnými zoomovacími gesty přibližovat a oddalovat podle toho, na co konkrétně se při animaci zaměřujeme.

### 4.1.2 Databáze MySql

**MySQL** je oblíbená relační databáze. Komunikace s ní probíhá pomocí jazyka SQL, resp. jedná se o dialekt tohoto jazyka. Je součástí většiny webových serverů.
Databáze obsahuje 3 základní tabulky, POOLSETTINGS eviduje sklady (regály), RESRCSETTINGS jednotlivé stroje a konečně tabulku krokování, což je vlastní časový plán.


Obr. 12: Schéma tabulek databáze ve vlastní webové aplikaci

### Sql dotaz

Níže uvedený SQL dotaz může na první pohled vypadat zbytečně složitě, na druhou stranu by mohl být poměrně dobře čitelný i po delší době, kdy může dojít ke ztrátě povědomí, co chtěl autor kódu tím kterým příkazem říct. Ke konci dotaz komplikujeme pomocí příkazu EXTRACT. Extrahujeme tím ze sloupce „Čas“ pomocné sloupce pro účely budoucího snadnějšího počítání v javascriptové části kódu, ověření, zda v dalším kroku (řádku) následuje druhý den (abychom mohli vyzvat uživatele např. vyskakovacím oknem, zda chce přeskočit noční dobu a neprodlužovat tak dobu běhu animace).

```
SELECT Tabulka AS typ, Polozka AS stroj,
(SELECT
CASE
    WHEN typ='resc' THEN (SELECT POZX FROM RESRCSETTINGS WHERE
NAME=stroj )
    WHEN typ='sklad' THEN (SELECT POZX FROM POOLSETTINGS WHERE
NAME=stroj )
END
) AS pozice_x,
(SELECT
CASE
    WHEN typ='resc' THEN (SELECT POZY FROM RESRCSETTINGS WHERE
NAME=stroj )
    WHEN typ='sklad' THEN (SELECT POZY FROM POOLSETTINGS WHERE
NAME=stroj )
END
) AS pozice_y,
ORDERID, LOADID, Status, Cas,
EXTRACT(MONTH FROM Cas) AS mesic,
EXTRACT(DAY FROM Cas) AS den,
EXTRACT(HOUR FROM Cas) AS hodina,
EXTRACT(MINUTE FROM Cas) AS minuta
FROM krokovani ORDER BY Cas ASC
```

Bylo by možné tabulku krokování rozšířit o sloupce pozice x-ové, y-ové souřadnice, typu položky (sklad, stroj). Tím by se sice zjednodušil SQL dotaz na jeden řádek, vybraly by se všechny položky (řádky) z tabulky, akorát by se ošetřilo setřídění podle času a výsledek dotazu bychom uložili do struktury pole. Na druhou stranu bychom tak popřeli základní databázový přístup, když bychom opakovaně kopírovali pozici x, y, a typ k příslušné položce objektu, mohlo by navíc dojít k nekonzistenci dat.

### 4.1.3 Three.js

Three.js představuje oblíbenou a poměrně rozšířenou JavaScriptovou knihovnu se širokou podporou webových prohlížečů, která slouží právě k vytváření a zobrazování 3D animací ve webovém prohlížeči. Používá rozhraní WebGL. Do webové stránky načtené 3D modely jsou tak akcelerované přes GPU (grafický procesor, anglicky graphics processing unit). Knihovna Three.js byla uvedena v roce 2010 Ricardem Cabellem zveřejněním na GitHubu. Rendrování (renderer) bylo začleněno později, s nástupem technologie WebGL, Paulem Bruntem. Knihovna Three.js je podporována všemi

moderními webovými prohlížeči, podporujícími WebGL technologii. Důležité pro volbu právě této knihovny je, že je dostupná pod licencí MIT.[13]

Celá knihovna je uložena v jediném JavaScriptovém souboru. Nicméně pro komplexnější animaci, načtení různých 3D modelů, drag&drop metody jsou dále používány rozšiřující skripty, vše nyní součástí aplikačního rozraní Three.js. [14]

**Načtení základních Three.js skritptů:**

```
<script type="module">

import * as THREE from '../build/three.module.js';

import { OrbitControls } from
"./jsm/controls/_OrbitControls.js"; //upravene

import { DragControls } from "./jsm/controls/_DragControls.js";
//upravene

import { ColladaLoader } from './jsm/loaders/ColladaLoader.js';

import { STLLoader } from './jsm/loaders/STLLoader.js';

import { IFCLoader } from './jsm/loaders/IFCLoader.js';

import Stats from './jsm/libs/stats.module.js';
```

### 4.1.4 Vytvoření 3D scény, kamery, pomocné mřížky:

```
function init() {

          const container = document.getElementById( 'container'
);

// scene

          scene = new THREE.Scene();

          scene.background = new THREE.Color( 'white' );

          scene.add( new THREE.AmbientLight( 0x999999 ) );

// camera

          camera = new THREE.PerspectiveCamera( 10,
          window.innerWidth / window.innerHeight, 1, 300 );


          camera.up.set( 0, 1, 0 );

          camera.position.set( -30, 120, 60 );

          camera.add( new THREE.PointLight( 0xffffff, 0.75) );

          scene.add( camera );
```

```
 // hlavni mrizka
          const grid = new THREE.GridHelper( 50, 20, 0xffffff,
0x555555 );
          grid.position.x = 18;
          grid.position.z = 0;
          scene.add( grid );
// render
          renderer = new THREE.WebGLRenderer( { antialias: true
} );
          renderer.setPixelRatio( window.devicePixelRatio );
          renderer.setSize( window.innerWidth,
window.innerHeight );

          container.appendChild( renderer.domElement );
}
```

### 4.1.5 Načtení 3D modelů

Pro zobrazení jednotlivých strojů a skladů (regálů) v navrhované aplikaci s výhodou kombinujeme více metod, resp. programovacích jazyků. Nejprve pomocí PHP komunikujeme s MySQL databází, kde máme uložené naše stroje a mezisklady. Jednoduchým SQL SELECT dotazem si nejprve uložíme všechny stroje do PHP pole,

```
$data_x = array();  $data_y = array();  $nazev_modelu = array();
$typ_stroje = array();
$query = "SELECT * FROM RESRCSETTINGS";
$result = mysqli_Query($conn, $query);
$j=0;
while ($data = $result->Fetch_Array ()){
    $data_x[$j]=$data['POZX'];
    $data_y[$j]=$data['POZY'];
    $nazev_modelu[$j]=$data['NAME'];
    $typ_stroje[$j]=$data['typ'];
$j++;
}
```

jehož obsah pak v JavaScriptové části kódu přeneseme do JavaScriptového pole, například:

```
var typStroje = <?php echo json_encode($typ_stroje) ?>;
```

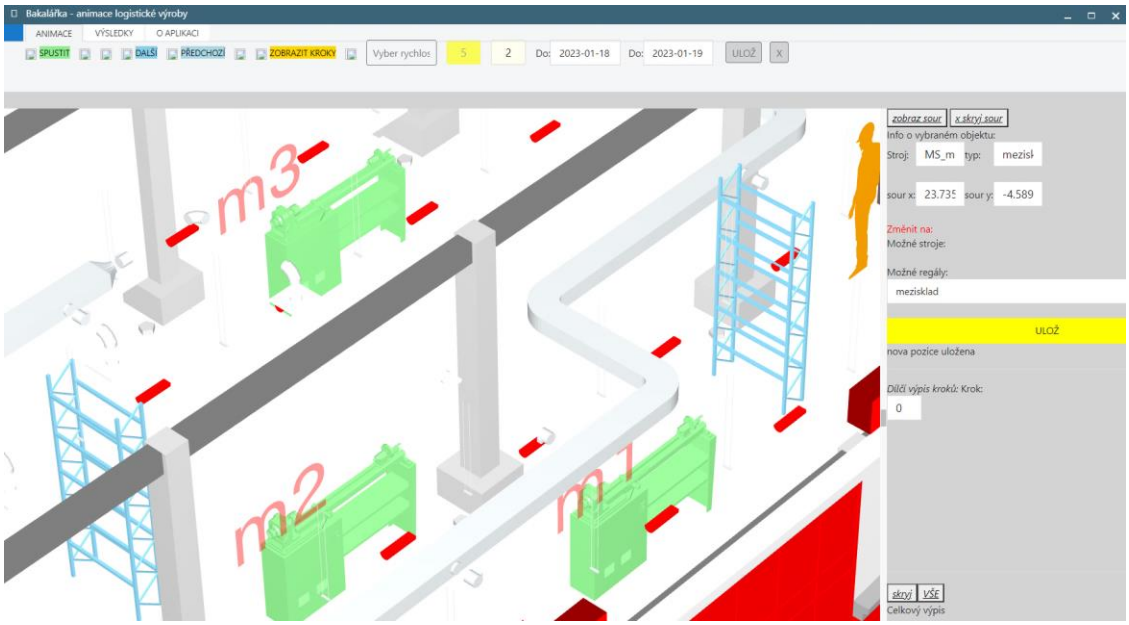
Knihovna Three.js umožňuje jednoduché načtení 3D modelu do webové stránky, a to například modely typu *.obj, *.dae (collada), *.stl, *.ifc.
Pro stroje a mezisklady byly zvoleny *.stl modely, a to hned z několika důvodů.
STL modely bylo jednoduché importovat jak do desktopové, tak webové aplikace, objekty je možné vytvořit a exportovat z volně dostupného a oblíbeného SketchUPu,
V rámci této práci nebyly tvořeny vlastní modely, avšak převzaty volně dostupné z [15]

```
let group_stroje, objects;
var souborStroje='./models/stl/binary/soustruh-binary.stl';
for(let i=0; i < xx.length; i++){
const loader2 = new STLLoader();
loader2.load( souborStroje, function ( geometry ) {
          const barva = new THREE.Color( 'lightgreen' );
              const material = new THREE.MeshPhongMaterial( {
                    color: barva , specular: 0x111111,
          shininess: 200 } );
                    let meshMaterial = material;
          const mesh = new THREE.Mesh( geometry, meshMaterial );
                    mesh.position.x = xx[i];
                    mesh.position.y = 0;
                    mesh.position.z = yy[i];
                    mesh.scale.set( 2, 2, 2  );
                    mesh.name=stroj[i];
                              mesh.typ=typStroje[i];
                    mesh.castShadow = true;
                    mesh.receiveShadow = true;
          group_stroje.add( mesh );
          objects.push( mesh );
          } );
} //konec cyklu načtení strojů
```

### 4.1.6 Rozmístění strojů a skladů na pracovní ploše

Po úspěšném načtení 3D modelů z databáze máme možnost přizpůsobit si pracovní resp. vizualizační plochu svému skutečnému stavu. Po dvojkliku na objekt stroje nebo regálu se v pravé části programu zobrazí pomocné informace o modelu. Souřadnice pozic x, y se vypisují automaticky během „addEventListener( 'drag', function( event ){“ metody typu „drag&drop“, zároveň se automaticky ukládá nová pozice do databáze pro ošetření případů, že uživatel pracně přeskládá stroje na pracovní ploše a zapomene si práci uložit. Mohla by být zde samozřejmě i naopak možnost, že v rámci snahy o přetočení či zoomování plochy nechtěně přesune stroj do jiné pozice a tím dojde k jejímu automatickému nežádoucímu uložení, tento případ však způsobí menší škody, lze načíst jednoduše znovu ze zálohy.

Pokud jsme nyní spokojeni s grafickým návrhem rozmístění strojů a skladů na pracovní ploše, můžeme přistoupit k vlastní animaci.




Obr. 13: Ukázka z vlastní webové aplikace

### 4.1.7 Animace

Po vyjasnění problematiky spojené s vlastní vizualizací pracovní plochy bylo nutné vymyslet způsob řešení hlavní části tématu, animace. V tomto případě bylo zvoleno použití animačního cyklu vepsaného přímo do funkce init()

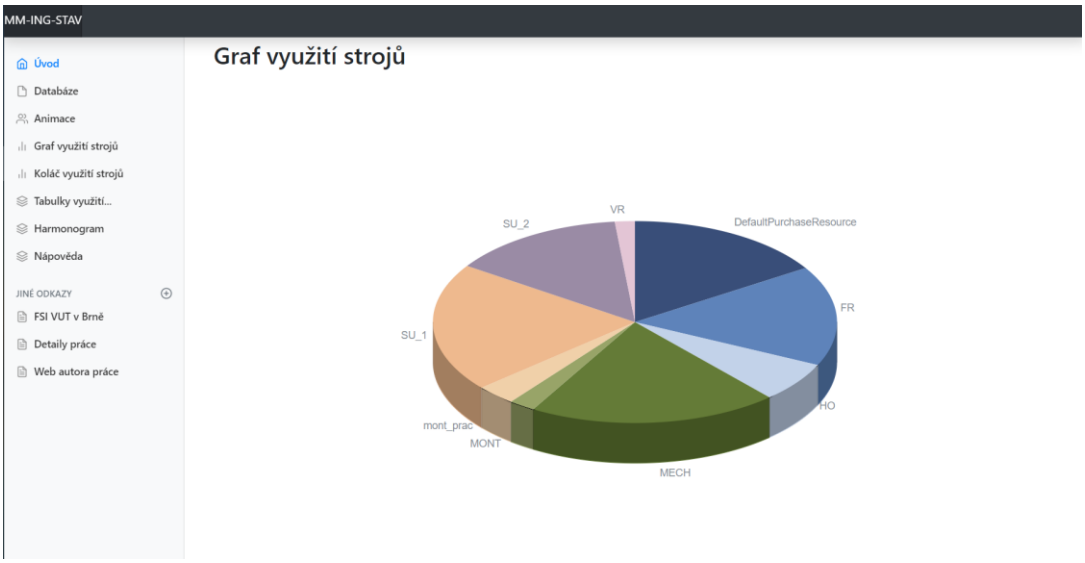
```
renderer.setAnimationLoop(() => {

// vlastní animační kód, cyklicky se opakující funkce, lze
zobrazit případně ze zdrojového kódu webové aplikace

}
```

### 4.1.8 Další funkce programu

Simulační a animační programy by však měly mít určitě zachovány i klasické výstupy plánování výroby, jako jsou komplexní tabulky časového plánu, harmonogram, grafy.




Obr. 14: Ukázka výstupu vlastní webové aplikace

### 4.1.9 Návrh vzhledu aplikace, použité šablony

Jelikož webovou aplikaci zobrazujeme na různých zařízeních, od velkého monitoru, přes běžný notebook po tablet nebo mobil, je žádoucí, aby webové stránky resp. naše aplikace se chovala tzv. responzivně, tedy aby se její obsah a zobrazení přizpůsobovalo velikosti zařízení, na kterém aplikaci prohlížíme. V případě, že autorem navržená optimalizace na jiné zařízení se nám nelíbí a nechceme si zvykat na různé rozložení ovládacích prvků na desktopu a jiné na mobilním zařízením, vždy máme možnost přepnout mobilní prohlížeč do režimu „zobrazení jako na PC“. Webové stránky se tak dnes většinou navrhují metodou „mobile first“, tzn. klademe větší důraz na návrh pro mobilní zařízení a poté pomocí CSS (kaskádových stylů) optimalizujeme na větší rozlišení.

U webových aplikací, a to zejména v tomto konkrétním případě, kdy stěžejní částí aplikace je 3D vizualizační plocha, responzivní chování a metoda „mobile first“ se neukázala jako vhodné řešení, proto bylo postupně od této metody upuštěno a od poslední verze má aplikace naopak snahu napodobovat klasický desktopový program a nevypadat jen jako webová stránka.

Pro alespoň částečné možnosti přizpůsobení menší zobrazovací ploše byly přidány některé funkce a různé tlačítka zobrazení / skrytí podrobnějších informací na bočním panelu, změna jeho velikosti splitter lištou, zobrazení a skrývání nepotřebných tlačítek během animačního režimu pro zmenšení velikosti horní lišty.

Stejně jako u většiny desktopových programů je dále žádoucí, aby bylo rozložení uživatelských prvků (nástrojové lišty, ovládací ikony) víceméně podobné jako u většiny

běžných programů, tak i při návrhu webových stánek a aplikací je snaha o zachování běžných standardů, jako horní nástrojová lišta, boční informační panel. Nejen proto, ale i z důvodu úspory času (a v neposlední řadě si přiznejme, že většina z nás techniků či programátorů nemá zrovna grafické cítění pro vytváření vlastního designu aplikace), se s oblibou využívá šablon a různých vývojových frameworků. Pro návrh této webové aplikace bylo kombinováno několik běžně používaných šablon, kdy nejznámější je zřejmě **Bootstrap** [16]. Postupným vývojem aplikace však všude přítomný Bootsrap styl přestal vyhovovat. Novým představám se začal lépe přibližovat framework **Metro 4** [17]. Z tohoto byl převzat zejména styl horního menu, pomocí „splitteru“ pak rozložení dvou hlavních panelů.

Při vývoji aplikace nebylo využíváno žádných grafických editorů (známé také pod zkratkou WYSIWYG, což je akronym anglické věty „What you see is what you get“, v českém překladu „co vidíš, to dostaneš“). Postupně bylo zjišťováno, že tento přístup není právě efektivní, vede často ke snaze kopírování již vytvořeného a vytváření tzv. „špagetového“ kódu tak typického pro webové stránky.

Pro budoucí vývoj webových aplikací se nyní jeví jako vhodnější nejprve provést průzkum možných frameworků. Nicméně v době vzniku této práce v oblasti webových frameworků nebyl žádný známý tak propracovaný, jako s oblibou využívaný grafický návrhář WinForms či WPF aplikací, dostupný pro programování klasických programů v MS Visual Studiu, kdy programátor grafický návrh své aplikace vytváří většinou přetahováním nástrojů z toolboxů a vývojové prostředí se pak stará o automatické vygenerování většiny kódu. Když už některé řešení zaujalo, bylo zavrženo z důvodu ztráty kontroly nad vlastním kódem, zejména v případě různých JavaScriptových frameworků. Při navrhování webových stránek se tak často používají šablony z jiných stránek, či dříve vytvořené „knihovny“ kaskádových stylů CSS, jako zmíněný Bootstrap.

## 4.2 DESKTOPOVÁ VERZE

Cílem této bakalářské práce však bylo také naprogramovat software pro animaci výroby. V této kapitole se tedy dostáváme k ukázce vlastního řešení klasického programu. Software je psán v programovacím jazyce C#, ve vývojovém prostředí (IDE) od Microsoftu, Visual Studio, ve verzi 2017. Programovací jazyk C# byl zvolen z důvodu, že v něm získal autor práce alespoň základní znalosti v rámci svého studia. Pro začínající programátory je zřejmě vhodnější, než například C++ či Visual Basic. V neposlední řadě bylo, možná jen ze subjektivního úhlu pohledu, jednodušší přes známé webové platformy jako StackOverflow.com, Codeproject.com, csharphelper.com nalézt k C# více návodů a vzorových ukázek kódu než v případě studia jiných programovacích jazyků.

Jako vývojové prostředí (IDE) bylo zvoleno zmíněné Microsoft Visual Studio, Enterprise, 2017 [18], konkrétněji z důvodu, že v praxi se s MS Visual Studiem zřejmě setkáme nejčastěji, v rámci studia máme toto IDE k dispozici zdarma, a shodná verze byla nainstalovaná ve studijních učebnách na fakultě. V předchozích ročnících bylo zjištěno, že pokud programátor pracuje v jiném prostředí doma na svém více zastaralém

počítači a v jiném v práci, na fakultě, přenositelnost kódu není vždy úplně zaručena a bylo téměř vždy nutné potýkat se s vyladěním pár chyb, které IDE hlásilo při otevření v novější verzi. Tento poslední důvod však bohužel ztratil na svém významu vzhledem k době, ve které tato bakalářská práce vznikala. Vzhledem k pandemickým opatřením nebylo bohužel buď vůbec možné, anebo pak už s určitými omezeními, navštěvovat fakultu.

### 4.2.1 WPF (Windows Presentation Foundation).

Při založení nového projektu ve Visual Studiu máme na výběr z několika možností. Byl zvolen **framework WPF** (Windows Presentation Foundation). Nabízelo se použití například staršího **Windows Forms,** nicméně vzhledem k požadovaného návrhu grafické aplikace je v dnešní době stále více prosazován WPF. Bylo by možné program navrhnout pouze s vestavěnými knihovnami pro WPF od MS Visual Studia, nicméně je ve vlastním řešení kombinováno více technologií.

Pro vizualizaci a vytvoření 3D scény byla nakonec zvolena externí knihovna **Helix Toolkit**. [19] Výsledné tabulkové výpisy jsou pak generovány prostřednictvím Windows Forms, jelikož s tímto prostředím měl autor v rámci studia alespoň nějaké zkušeností na rozdíl od žádných v případě WPF. Ze subjektivního pohledu je grafický návrhář využívaný ve starších Windows Forms aplikacích přehlednější, editace vlastností vložených elementů rychlejší. Grafické možnosti a vlastní stylování ve WPF je však opravdu bohatší a pro tuto konkrétní práci vhodnější.

### 4.2.2 Helix Toolkit

Jak již bylo naznačeno, pro vytvoření 3D scény a zobrazení modelů strojů bylo zvoleno využití sady programovacích nástrojů **Helix Toolkit.** Program tak přebírá strukturu Helix Toolkit vzorového příkladu dostupného z [19], nicméně postupně docházelo k vlastnímu upravování, případně kombinování různých metod od Helix Toolkitu, až z původně zmíněného příkladu zbyla vlastně jen stromová souborová struktura.
Zmínili jsme, že není nutné využívat externí knihovny pro 3D vizualizaci a animaci, nicméně pro manipulaci s 3D scénou, natáčení, zoomování, drag&drop metody přesouvání 3D modelů je zřejmě stejně nutné standardní WPF kód doplnit o externí knihovny. Helix Toolkit byl vybrán zejména z důvodu, že autora ještě před touto bakalářskou prací zaujala problematika BIM, Building Information Modeling, což je zjednodušeně řečeno nový přístup k projektování a realizaci staveb ve stavebnictví, což je původně vystudovaný obor autora práce.

Níže je ukázka zdrojového kódu WPF okna aplikace, je psáno deklarativním jazykem XAML, což je odlišnost od generace kódu WinForms. I v tomto případě dochází k automatickému generování kódu pomocí grafického návrháře, který je oproti WinForms subjektivně méně propracovaný, na druhou stranu upravovat lze oproti WinForms jednodušeji i prostým psaním, či kopírování kódu ručně, který je jak vidět svojí strukturou podobný HTML formátu. Většina programátorů začíná právě návrhem

webových stránek, i proto WPF aplikace mohou být v konečném důsledku oblíbenější pro většinu programátorů než právě WinForms.

```
<Window x:Class="ExampleBrowser.Window1"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation
"
       xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
xmlns:mc="http://schemas.openxmlformats.org/markup-
compatibility/2006"
xmlns:local="clr-namespace:ExampleBrowser"
xmlns:System="clr-namespace:System;assembly=mscorlib"
             mc:Ignorable="d"
xmlns:helix="clr-
namespace:HelixToolkit.Wpf;assembly=HelixToolkit.Wpf"

Title="MM_Bakalářka_Animace výroby_FSI VUT v Brně, 2020/2021"
Height="547" Width="1479" WindowStartupLocation="CenterScreen"
WindowState="Maximized">

<Window.DataContext>
<local:ShellViewModel/>
</Window.DataContext>
<DockPanel Margin="0,-1,0,0">
<Menu DockPanel.Dock="Top" Background="#FFB4D3E4" FontSize="14" >
      <MenuItem Header="Soubor">
        <MenuItem Header="Načti data" Click="Nacti_Click"/>
        <MenuItem Header="Uložit plochu" Click="Ulozit_Click"/>
           <Separator/>
```

### 4.2.3 Popis návrhu a ovládání vlastního programu

Po spuštění programu i v tomto případě nejdříve naběhne jen úvodní obrazovka, která pouze slouží k informování, že se jedná o program k bakalářské práci VUT a obsahuje jediné tlačítko „POKRAČOVAT“.

Program není nutný instalovat, pouze se rozbalí ZIP archiv. Ke spuštění programu by neměla být vyžadovaná ani žádná administrátorská práva. Pokud se tedy čtenář rozhodne pokračovat, s vědomím vlastního rizika, že se jedná o verzi programu uvolněnou v rámci odevzdání teprve bakalářské práce, dostane se do hlavního okna, které se zobrazí automaticky už maximalizované na celou obrazovku. Jak bylo zmiňované dříve, je žádoucí, aby programy a aplikace měly běžně zažitý design a rozložení ovládacích prvků. Proto i v tomto případě je zde klasické horní menu a nástrojová příkazová lišta. Použitím entity toolbar je zajištěno, že v případě zmenšení okna se tlačítka, které se nevešly do zobrazené plochy, skryjí, avšak s možností rozbalení vysouvacím seznamem. Je dobrým zvykem, že příkazy z nástrojové lišty jsou dostupné

i z horního menu. To je zde samozřejmě dodrženo. Aby nedocházelo ke zdvojování stejného kódu na dvou místech, vytvoří se samostatně funkce, která je pak volána buď z obslužné metody tlačítka na nástrojové liště, anebo z položky horního menu.

Po úvodním seznámením se s oknem programu přejdeme zřejmě k načtení zdrojových dat. Zdrojovými daty je myšleno otevřít soubor, který obsahuje ve vhodné struktuře seznam strojů, meziskladů, které se mají zobrazit na vizualizační ploše. Na výběr je ze dvou možností. Databáze MS Access, anebo sešit MS Excel. Vzhledem k tomu, že je potřeba, aby program například věděl, který list Excelu obsahuje seznam strojů, který seznam skladů, v jakém sloupci se nachází x-ová souřadnice polohy modelu a v jakém je naopak y-ová, je logické, že animace bude fungovat pouze za předpokladu, že načtená zdrojová data budou v předepsané struktuře. S programem jsou v instalační složce i vzorové soubory, excelový sešit a Access databázový soubor.

Výběr těchto dvou formátů má své opodstatnění. Jako první byl zvolen Accessový soubor s myšlenkou, že program bude sloužit jako řekněme nástavba k vyučovanému programu Factor/AIM. V průběhu studia problematiky však bylo zjišťováno, že většina plánovacích programů umožňuje export do Excelu a v neposlední řadě se jednalo o požadavek nebo doporučení vedoucího práce o umožnění načtení dat přímo z excelového sešitu, bez nutnosti nějakého mezikroku uživatele, jako je přeuložení do jiného formátu, import do Accessu apod. Vedoucímu práce bylo nakonec dáno za pravdu, protože MS Access není stále standardně používaným kancelářským programem, nejen z neznalosti databází běžného uživatele počítače, avšak i vzhledem k faktu, že MS Access není součástí základního kancelářského balíčku MS Office. Když se nad tím zamyslíme, je to celkem škoda a možná právě i proto je většinou uživatelů stále na vše používán Excel. Lze namítnout, že jsou k dispozici zdarma ke stažení i alternativy k MS Office, jako LibreOffice (dříve OpenOffice), ve kterých už databázová aplikace je standardně součástí. Přiznejme si však, že většina uživatelů má prostě od začátku k dispozici předinstalovanou verzi MS Office, takže jiné alternativy nemá příliš důvod vyhledávat.
To jsme ale mírně odbočili od vlastního tématu. Abychom tedy načetli zdrojová data, klikneme v nástrojové liště na první tlačítko **„Načti data“.** Otevře se dialogové okno, ze kterého vybereme svůj soubor. Aby uživatel nemohl vybrat jiný formát souboru a pak se nedivil, že něco nefunguje, anebo že na něj vyskakuje varovné hlášení o chybě, je přímo ve zdrojovém kódu obslužné metody dialogového okna provedeno vyfiltrování pouze konkrétního typu přípony souboru.

Pro Access soubor a Excel sešit je samostatná ikona resp. položka v menu. Bylo by možné řešit jedním tlačítkem, a podle vybraného typu by se vyvolala příslušné funkce, nicméně i takové aplikace jako Word mají pro načtení zdrojových dat různá tlačítka, proto je tomu tak i v tomto případě pro zachování určitého běžného standardu.

Načtení 3D modelů strojů a skladů si vyžádá určitý čas, podle výkonu počítače, nicméně bylo zkoušeno na dnes již morálně zastaralém počítači s RAM pod 3GB. Pro jistotu je však v kódu omezení na 100 strojů. Přestože jsme uživatele v úvodu seznámili s tím, proč musí být zdrojová data v naší struktuře, nedonutíme uživatele, aby používal naše názvy strojů. Pokud se tedy název stroje či meziskladu neshoduje s přednastaveným

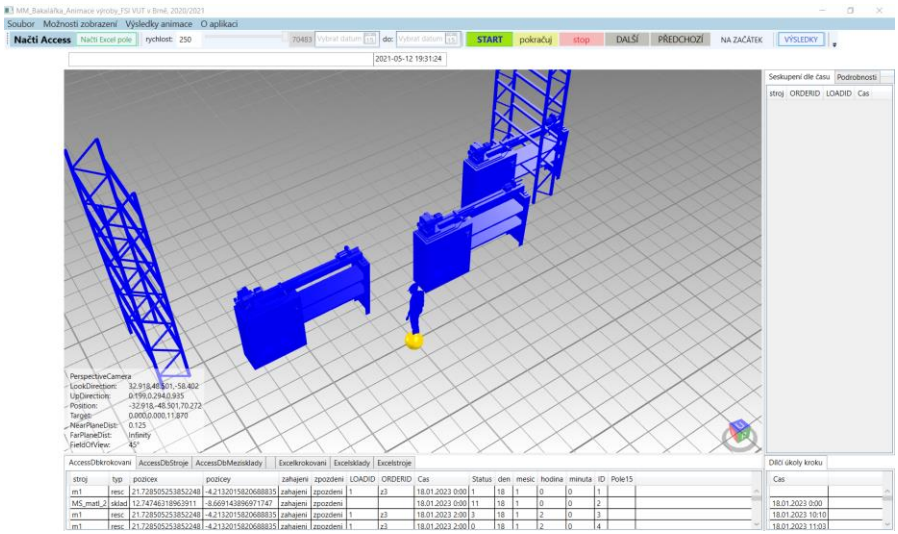
názvem, ke kterému je definován konkrétní 3D model, defaultně se zobrazí jako stroj model soustruhu a jako mezisklad regálový systém.

Při prvním načtení zdrojového souboru, vlastního časového harmonogramu v Excelu anebo exportu z jiného plánovacího programu, může dojít k nutnosti změny rozložení strojů po pracovní ploše. To jsou pouze kosmetické úpravy, nicméně pro výsledný efekt, uživatelský komfort a srovnání se s ostatními aplikacemi nezbytnost.
Je tedy umožněno vlastní přizpůsobení pracovní plochy, jako je:

- posun, rozložení prvků metodou drag&drop
- otočení modelů
- změna měřítka modelu.
- změna 3D modelu z předvybraných knihoven prvků (soustruh, pila, CNC).

Co se týká přednastavených typů 3D modelů, je k dispozici načtení pouze *.stl modelu. Zdůvodnění této volby bylo již v části webové aplikace, jedná se mimo jiné o dostupný formát přímo z neplacené verze oblíbeného Google SketchUPu, bez žádného dalšího konvertování. Ve webové aplikaci byla přidána i možnost naimportování vlastní haly ze souboru *.ifc, což je běžný standard ve stavebnictví pro 3D modely staveb, nicméně zde tato možnost nebyla přidána. V případě, že uživatel chce pro zkrášlení výsledného efektu vložit model haly, doporučuje se jednoduché předkreslení právě ve SketchUPu, anebo nějakým jiným způsobem překonvertovat *.ifc soubor, 3D stavební model haly, do *.stl formátu.

Přidání libovolného počtu strojů je teoreticky možné přidáním řádku do tabulky, databáze, nicméně bez doplnění řádků i v krokovací tabulce nemá přidání stroje žádný efekt. Zachován by tak měl být princip, že tato aplikace slouží jako grafická 3D nástavba na jiný plánovací program případně vlastní Excel a změny provádět výhradně tam. Po provedení změn a vypočítání nového časového plánu pak re-importovat do našeho programu. Algoritmy plánování nejsou a neměly být předmětem tohoto programu, této bakalářské práce. Přidání algoritmů plánování přímo do našeho programu je určitě žádoucí a námět na případný další rozvoj a studium.




Obr. 15: Ukázka obrazovky po načtení zdrojových dat

Pokud jsme tedy nyní spokojeni s rozmístěním strojů po pracovní ploše a blíží se tak realitě, přistoupíme k vlastními popisu spuštění a ovládání animace.

### 4.2.4 Rychlost animace

Jako první nás bude zajímat rychlost animace. Při prvotním načtením strojů došlo na pozadí i k vytvoření časového pole, což je seskupení krokovací tabulky / listu podle sloupce času zahájení. Z takto seskupených skutečných časů se do pole uloží také doby, počítané od počátečního data/času animace, ve kterých má dojít k načtení další skupiny kroků.

Uživatel si zvolí rychlost animace z několika nabízených možností:

- vepsání do TextBoxu v horní nástrojové liště,
- nebo plynulým výběrem pomocí „slideru“,
- vepsáním celkové délky doby animace, jak dlouho chce, aby na počítači běžela.

V tomto posledním případě dostane např. v dialogovém okně informaci, jak dlouhý časový úsek ve skutečném světě vybral k animaci a podle toho zváží, jak dlouho chce, aby mu na počítači běžela animace. Tato možnost bude přidána z důvodu, aby uživatel nemusel volit rychlost animace metodou „pokus - omyl“, jak tomu běžně bývá zvykem. V backendu programu se podle zadaných údajů vepíšou do časového pole hodnoty, ve kterém okamžiku (od zahájení animace, v milisekundách) má dojít k přesunu k dalšímu animačnímu kroku, k další skupině úkolu v tom určitém čase.

Tento na první zamyšlení nadbytečný úkon je zahrnut do kódu pro ošetření případů, že v jednom čase, např. v 8:00 ráno, je zahájeno více úkonů současně – seřízení jednoho stroje, práce druhého, pohyb dávky ze třetího na mezisklad apod. Kdybychom procházeli řádek po řádku, úkon po úkonu, docházelo by k vytváření nadbytečných kroků, respektive nutných kliků v případě procházení animace přeskakováním přes tlačítko další.

Dalšími možnostmi optimalizace, výběru rychlosti jsou:

- pohyb dávek mezi stroji – zadá se podobně, jakou rychlostí (pro změnu například výběrem pomocí SelectBoxu) se má na obrazovce animovat a program podle vzdálenosti mezi jednotlivými stroji spočítá dobu, která se má dosadit do metody DoubleAnimation, která bude popsána dále.
- doba prodlevy mezi jednotlivými kroky „vnitřního pole“ – tato možnost byla přidána v rámci ošetření případu, kdy zdrojová data od uživatele obsahují různé činnosti u stejného stroje ve shodný okamžik, např. v 7:30 je u stroje „m1“ naplánováno seřízení stroje, ale zároveň v 7:30 i zpracování produktu u stroji. V zadané zdrojové tabulce nebyly od uživatele známé časy, jak dlouho má nejprve trvat seřízení stroje. Proto dostaneme uživatel při animaci možnost volby, jak dlouho chce takto neošetřené doby činností animovat. Defaultně bude nastavené např. na 0,5 s.

### 4.2.5 Spuštění animace, optimalizace časovače Timer

Po výběru rychlosti a případně časového úseku od / do už nezbývá nic jiného, než spustit animaci tlačítkem „Start“. Spustí se „krokovač“, který je upraven tak, aby v časové třídě „Timer“ nedocházelo vzhledem k zatížení systému ke zkreslování animačního času. To byl hlavní problém zajištění plynulé animace.

Úprava spočítá v tom, že se zjišťuje rozdíl času (v sekundách nebo lépe v milisekundách) od doby kliknutí na start. Nesmí se v „krokovači“, chceme-li v „Timeru“, používat prostá inkrementace (zvyšování čísla o nějakou hodnotu). Podle výkonu počítače resp. podle právě prováděných operací pohybu dávek, dělníků, dochází ke zpožďování dalšího volání metody „timer_Ticku“ oproti zadanému intervalu. Nejde tak s jistotou říct, že další skupina kroků nastane právě tehdy, když velikost timeru je na určité hodnotě.

V řešení přes webovou aplikaci s pomocí Three.js se s výhodou využilo animačního cyklu, kde také příliš nedochází k ovlivňování rychlosti cyklické třídy podle zatížení systému. Ve WPF aplikacích v době tvorby programu nebyla vhodná třída animačního cyklu nalezena, přestože jistě musí existovat. Pomocí „DateTime.Now“ ověřování aktuálního času resp. jeho rozdílu tak vlastně došlo k vytvoření vlastní animační cyklické funkce, zřejmě téměř shodně fungující.

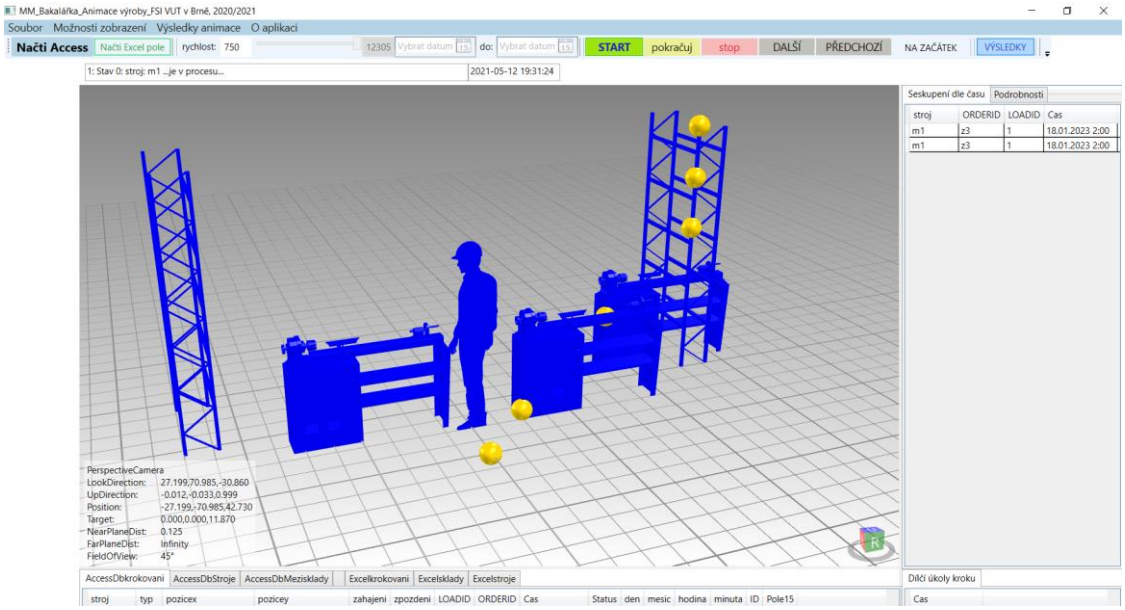
Jakmile tedy buď automaticky pomocí časovače, nebo manuálně přes krokovací tlačítko „další“, dojde k přesunu na další krok, inkrementuje se o jedna pomocná proměnná „krok“, vytvoří se stejně jako u JavaScriptové verze kódu řekněme vnitřní krokovací pole.

V prvním případě webové aplikace jsme prošli všechny položky vnitřního pole naráz přes cyklus for a podle jistého stavu položky poslali jako nový záznam buď do dalšího pole, listu položek „v pohybu“, nebo provedli jinou danou akci.

V tomto případě chceme projít položky vnitřního pole postupně. Jelikož nebyla v zadání dána rychlost kroků, které se mají provést ve stejný čas, bylo zvoleno, že budou za účelem postřehnutí během animace od sebe opožděny o 500 ms. Bude přidána možnost rychlost uživatelsky měnit. Nejdříve bylo cílem bylo vyhnout se možná subjektivně neoblíbenému systémovému „Timeru“. Na druhou stranu se ale ukázalo jeho použití v tomto případě nejvhodnější. Vzhledem k tomu, že účelem tohoto druhého časovače je pouze záměrné animační zpoždění dalšího kroku, není případné zdržení časovače kvůli zatížení stroje významné. Použití časovače se ukázalo jako nevhodné pouze v případě, že se měl použit pro inkrementaci a podle hodnoty proměnné provést volání nějaké další funkce v konkrétním okamžiku. Vzhledem ke zdržení inkrementace předchozími úkony by nebylo správného, včasného krokování dosaženo.

### 4.2.6 Pohyb dávek.

Dalším problém, který si zaslouží podrobnější popis, je pohyb dávek. Dávkami v tomto případě rozumíme rozpracované výrobky, či skupinu výrobků. V rámci animace zjednodušeně zobrazujeme jako žlutou krychli nebo kouli (lze si představit jako úložný box, krabici, do které obsluha stroje uloží rozpracované nebo hotové výroky a pošle je na dopravníkový pás nebo donese osobně ke zpracování dalšímu stroji).




Obr. 16: Ukázka pohybu dávek, postupné ukládání od strojů k meziskladům

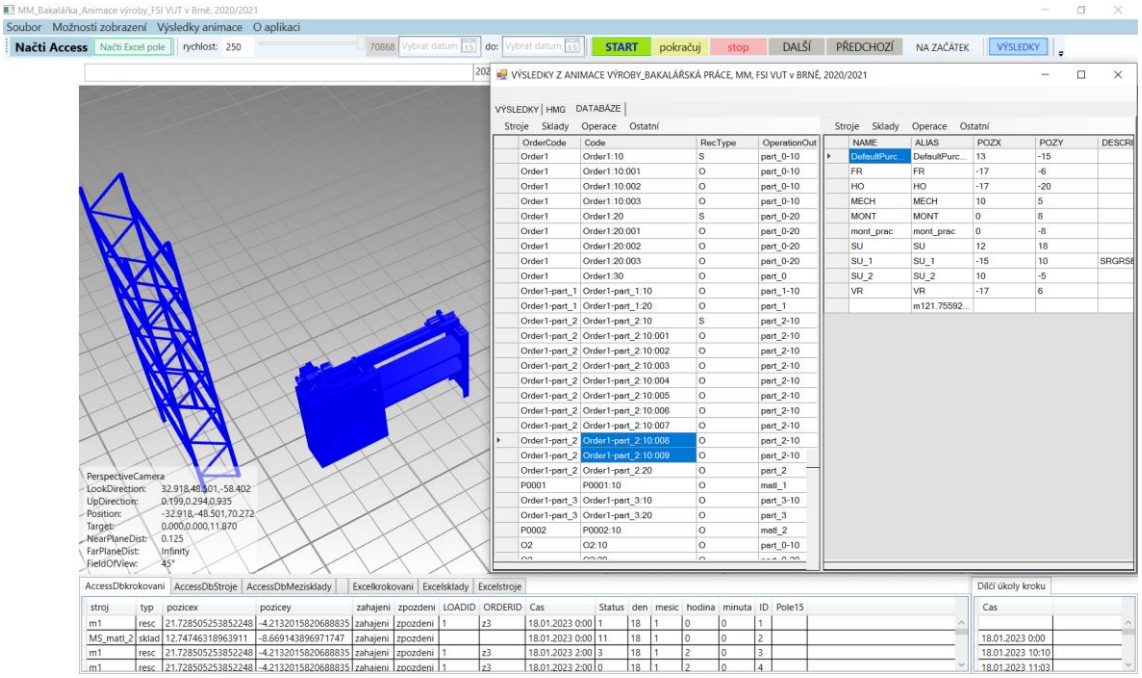
Ve webové aplikaci jsme se záměrně vyhnuli použití další externí knihovny v podobě např. TWEEN.js a vytvořili jsme si vlastní funkci pohybu, kdy se neustále prochází pole položek, které se mají dát „do_pohybu“, ověřuje se jejich x-ová souřadnice vzhledem k pozici cílového objektu, po jeho dosažení dojde k pohybu v druhém směru, stejnou logikou. Tím je pohyb dostatečně pod kontrolou programátora.

V tomto případě bylo využito, že ve WPF aplikacích existuje třída „**DoubleAnimation**“ (obor Systém.Windows.Media.Animation).
Animuje hodnotu Double, hodnotu mezi dvěma cílovými souřadnicemi, pomocí lineární interpolace během zadané doby. Tím je zajištěn plynulý pohyb odkud, kam. Dále je zadáno, jak dlouho pohyb potrvá. Zároveň se vyhneme ve WPF aplikací poměrně nespolehlivě resp. nepředvídatelně fungující metodě časovače Timer, závislého na zatížení systému.

Jinou možností, jak ve WPF aplikacích zajistit animaci, je pomocí vlastnosti „**StoryBoard“**. S tímto způsobem se případně seznámíme později, v dalším studiu, pokud u této zajímavé problematiky 3D animací a vizualizací zůstaneme.
Tím jsme úspěšně prošli přes většinu nástrah spojených s tímto způsobem animace a vizualizace dat.

### 4.2.7 Výstupy z aplikace:

Kliknutím na tlačítko „VÝSLEDKY“ se otevře samostatné okno s formulářem, ve kterém se do „DataGridView“ tabulek zobrazují tabulkové výpisy výsledků a lze číst údaje ze vstupních zdrojových dat i přímo v aplikaci. Výsledky, respektive náhled do části databáze se používá v případě výběru zdrojových dat z Accessu, jelikož ne všichni uživatelé disponují právě MS Accessem. V případě zdrojových dat z Excelu se doporučuje pro analýzu výsledků respektive analýzu zdrojových dat použití právě MS Excelu.




Obr. 17: Ukázka zobrazení výsledků resp. zdrojových dat

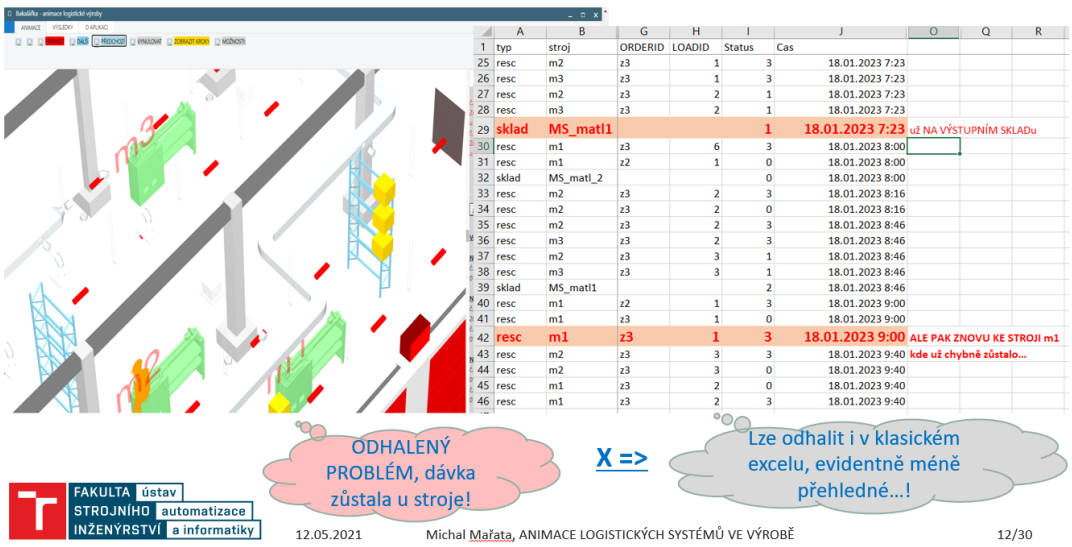
# 5 ZHODNOCENÍ

V této bakalářské práci tedy bylo cílem navázat na poskytnutý program z diplomové práce [12] od Ing. Veselského z roku 2012, vedené stejným vedoucím práce, program však přizpůsobit aktuálním potřebám zákazníků, a to zejména vizualizací ve 3D režimu.

V průběhu vývoje aplikace samozřejmě vyvstaly nějaké doplňující otázky a požadavky, jako například import přímo souboru „excelového“ typu, namísto původně uvažované MS Access databáze apod. Nebyl prováděn žádný průzkum trhu a ověřování, jakým způsobem menší výrobní společnosti plánují svou výrobu, nicméně lze usuzovat, že veškeré plánování se často omezuje na tabulkové procesory typu MS Excel.

Toho využívá i řešení popisované v této práci, kdy jednoduchým importem (nutně mírně upraveným do před-definované struktury), lze za využití stávajících, ve firmě zažitých plánovacích postupů, získat nástroj pro 3D animaci průběhu výroby.
Přes vzniklá úskalí při psaní programového kódu pro vlastní animaci se podařilo vytvořit poměrně života schopnou webovou aplikaci, která by mohla zaujmout zejména právě menší výrobní společnosti, stejně tak nakonec i požadovaný klasický .*exe program.
Jako ukázka a zhodnocení, zda průběh animace spuštěné v tomto programu může odhalit případné chyby v časovém plánu vytvořeném v tabulkové procesoru, slouží následující popisovaná část této kapitoly.

Na konci animaci by uživatel předpokládal, že všechny boxy (dávky) budou nějakým způsobem zpracovány, zejména lze i bez analýzy vstupních tabulek časového plánu usuzovat, že se dávky nezaseknou u některého stroje a budou na konci vyskládány na výstupním meziskladu (regálu). V našem případě testovacího souboru animovaných dat je vidět, že dvě dávky zůstanou na konci animace u stroje, což zřejmě nebyl záměr plánovače výroby. Že by tomu tak nemělo být si lze všimnout například znovu spuštěním animace, případně jejím zpomalení, kdy se u určitém čase dávky na výstupní mezisklad skutečně dostanou, jenže pak jsou v dalších z kroků opět volány na „scénu“ ke strojům.




Obr. 18: Ukázka odhalení možného problému ve zdroji dat na konci animace

Na důkaz, že se nejedná jen o chyby animačního cyklu, je připojena i vstupní tabulka dat s vyznačením předpokládané popisované chyby v plánování tohoto výrobního cyklu. Zde se jedná o testovací soubor dat se záměrně vytvořenou chybou, v praxi však může tato situace nastat jednoduše úplně stejně, zejména při používání k plánování výroby oblíbeného tabulkového procesoru. Naimportování dat do naší aplikace by tak mělo takové evidentní chyby, kdy dávky zůstanou ke konci animace u strojích, jednoduše odhalit. Složitější chyby, které se neukážou na konci běhu programu, ale jen v jeho průběhu, je nutné samozřejmě odladit jinými pozorovacími dovednostmi, ideálně v kombinaci s analýzou vstupních dat v podobě tabulkových výpisů, případně podpořenými i analýzou výstupů v grafech.

# 6 ZÁVĚR

Zadaným cílem této bakalářské práce byl popis a porovnání systémů pro animaci logistických systémů ve výrobě, čemuž se věnuje kapitola **3 Přehled současných obchodních řešení.** Z uvedeného výčtu lze nejvíce doporučit zřejmě program **Tecnomatix Plant Simulation** od **Siemens,** v praxi poměrně využívaný, i vzhledem ke známé značce společnosti, podpoře pro studenty, velice doporučovaný program.

Za zmínku však stojí i méně známý program **FlexSim**, který se jeví jako vhodně kombinující uživatelskou jednoduchost a přitom robustnost softwaru.

Pro většinu menších firem by však mohlo být zajímavé řešení navrhované v této bakalářské práci. Podařilo se vytvořit jak klasický desktopový software, spustitelný v prostředí Windows, tak navíc i webovou aplikaci, která umožňuje menším podnikům prezentaci postupného průběhu svých produktů výrobním procesem za pomocí 3D animace, přičemž zdrojem dat pro animaci je jednoduchá excelová tabulka.

Případní klienti, uživatelé této aplikace, tak nemusí přecházet na jiné robustní plánovací systémy a mohou nadále využívat své starší programy, většinou však neumožňující 3D vizualizaci. Je velmi pravděpodobné, že jejich stávající programy mají možnost exportu vytvořeného časového plánu produkce do souboru tabulkového procesoru, případně že firmy přímo pracují na rozvrhování své výroby například právě v MS Excelu.

Snahou a hlavním vytyčeným cílem tedy bylo vytvoření řekněme nástavby na programy typu Factor/AIM nebo z nich odvozené varianty, což se poměrně zadařilo a implementace tohoto řešení do programu podobného typu mu dodá na atraktivitě a použitelnosti v dnešní praxi. Díky struktuře zdrojových dat je možné a žádoucí další rozšiřování aplikace, například připojením na stávající zákazníkovi systémy a databáze, jako je objednávkový a fakturační systém, databázi skladových zásob apod. Tím lze z této aplikace, sloužící výhradně na animaci výroby, postupně vytvořit poměrně propracovaný software, vytvořit tzv. 4D aplikaci (čtvrtý rozměr značí čas, harmonogram výroby), nebo 5D software (pátý rozměr představuje ceny, doplnění o napojení na databázi cen, nákladů firmy).

Další možností, vhodnější, by bylo programovat aplikaci v takovém vývojovém prostředí resp. programovacím jazyku, ze kterého lze snadněji kompilovat na různé verze operačních systémů, a tedy udržovat pouze jeden kód. Nabízelo se tak zvolit i herní engine Unity, nicméně s vývojem v tomto prostředí nemám dostatečné zkušenosti, a tak bylo přistoupeno ke kombinaci dvou samostatných programů, aplikací.

Vytvoření desktopové verze, psané v MS Visual Studiu, za použití programovacího jazyka C# a knihoven HelixToolkit a webové aplikace využívající WebGL technologie, JavaScriptové knihovny Three.js, na kterou je soustředěna větší pozornost ze subjektivního důvodu větší obliby této technologie. V obou případech se jedná o naprogramování grafického uživatelského prostředí nad databází, takže změna logiky v obou verzích je snadná díky přenositelnosti stejných SQL dotazů, anebo díky podobné konstrukci polí, ve kterých jsou uchovávány data pro postupné kroky animace.

# 7 SEZNAM POUŽITÉ LITERATURY

[1] SystémOnLine. [online]. 2021 [cit. 2021-05-21]. Dostupné z:

https://www.systemonline.cz/clanky/simulace-a-optimalizace-v-planovani-vyroby.htm

[2] KONEČNÝ, Štěpán. Návrh a simulace výrobních systémů za využití simulačního softwaru pro pokročilé plánování a rozvrhování. Brno, 2020. Dostupné také z: https://www.vutbr.cz/studenti/zav-prace/detail/125508 . Bakalářská práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav automatizace a informatiky. Vedoucí práce Ivana Hromková.

[3] Siemens Industry Software Inc. [online]. 2021 [cit. 2021-05-11]. Dostupné z: https://www.konstrukter.cz/siemens-plm-software-vydal-novou-verzi-tacnomatixu/

[4] SEKEROVÁ, Tereza. Analýza a optimalizace výrobního systému pomocí počítačové simulace. Brno, 2014. Dostupné také z: https://core.ac.uk/download/pdf/30293731.pdf Bakalářská práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav automatizace a informatiky. Vedoucí práce Ivana Simeonová.

[5] Siemens Industry Software Inc. [online]. 2021 [cit. 2021-05-11]. Dostupné z: https://www.plm.automation.siemens.com/global/cz/products/manufacturing-planning/factory-line-design.html

[6] AXIOM TECH s.r.o. [online]. 2021 [cit. 2021-05-11]. Dostupné z: https://www.axiomtech.cz/25357-texnomatix-plant-simulation

[7] FlexSim Software Products, Inc. [online]. 2021 [cit. 2021-05-11]. Dostupné z: https://www.flexsim.com/flexsim/#3d-simulation

[8] ABB, s.r.o. [online]. 2021 [cit. 2021-05-11]. Dostupné z: https://www.therobotreport.com/abb-robotstudio-software-optimizes-robot-setups-with-videos/

[9] Autodesk [online]. 2021 [cit. 2021-05-11]. Dostupné z: https://www.autodesk.com/products/factory-design-utilities/overview?term=1-YEAR

[10] Lanner Group Limited. [online]. 2021 [cit. 2021-05-11]. Dostupné z: https://www.lanner.com/en-us/technology/witness-simulation-software.html

[11] The AnyLogic Company [online]. 2021 [cit. 2021-05-11]. Dostupné z: https://www.anylogic.com/manufacturing/

[12] VESELSKÝ, Josef. Animace výrobních systémů. Brno, 2012. Dostupné také z: https://www.vutbr.cz/studenti/zav-prace/detail/52011?zp_id=52011 . Bakalářská práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav automatizace a informatiky. Vedoucí práce Simeon Simeonov.

[13] Three.js. JavaScript Library [cit. 2021-05-11]. Dostupné z: https://cs.wikipedia.org/wiki/Three.js

[14] Three.js. JavaScript Library [online]. 2021 [cit. 2021-05-11]. Dostupné z: https://threejs.org/

[15] Dostupné z: https://3dwarehouse.sketchup.com/collection/a3498651-0c43-48d2-9b6b-9ba6565a2028/Industrial-machines

[16] Bootstrap. Web Framewok, CSS. [online]. 2021 [cit. 2021-05-11]. Dostupné z: https://getbootstrap.com/docs/5.0/examples/dashboard/

[17] Metro 4. Web Framewok, CSS [online]. 2021 [cit. 2021-05-11]. Dostupné z: https://metroui.org.ua/intro.html

[18] Microsoft. Visual Studio [online]. 2021 [cit. 2021-05-11]. Dostupné z: https://visualstudio.microsoft.com/cs/vs/

[19] HelixToolkit. Library for C# [online]. 2021 [cit. 2021-05-11]. Dostupné z: https://github.com/helix-toolkit/helix-toolkit

# 8 SEZNAM ZKRATEK, SYMBOLŮ A OBRÁZKŮ

## 8.1 Seznam použitých zkratek

| | |
|---|---|
| APS | angl. Advanced Planning and Scheduling Systems |
| CSS | angl. Cascading Style Sheets |
| HTML | angl. Hyper Text Markup Language |
| IDE | angl. Integrated Development Environment |
| MIT | angl. Massachusetts Institute of Technology |
| MS | angl. Microsoft |
| PHP | angl. Hypertext Preprocessor |
| SQL | angl. Structured Query Language |
| WebGL | angl. Web Graphics Library |
| WPF | angl. Windows Presentation Foundation |
| XAML | angl. eXtensible Application Markup Language |

## 8.2 Seznam obrázků

# 9 SEZNAM PŘÍLOH

Příloha A: *.zip archiv DEBUG složky

# PŘÍLOHY

Elektronicky odevzdaný ZIP archiv DEBUG složky:

- spustitelný *.exe program
- nezbytné *.dll knihovny
- vzorová databáze *.mdb
- vzorové zdroje dat v *.xlsx